Low-Complexity Iterative Multiuser Detection and Decoding for Real-Time Applications

Suman Das, *Member, IEEE*, Elza Erkip, *Member, IEEE*, Joseph R. Cavallaro, *Senior Member, IEEE*, and Behnaam Aazhang, *Fellow, IEEE*

***Abstract*—This paper presents a low-complexity multiuser decoding technique that can be implemented in real time for a convolutionally coded direct sequence code division multiple access (DS-CDMA) system. The main contribution, denoted here as the iterative prior update (IPU), consists of iterative interference cancellation and prior updates on sequences of coded bits combined with M-algorithm and list decoding. We illustrate performance gains over other low-complexity sequence detection and decoding strategies and argue that the algorithm converges within a few iterations and requires only a small size buffer for keeping track of the priors along iterations. The fact that the we can use existing available architectures for Viterbi decoding with slight modifications and can meet the real-time processing constraints makes the IPU algorithm an attractive alternative for cellular systems.**

***Index Terms*—Iterative detection and decoding, list decoding, multiuser detection and decoding.**

I. Introduction

THIS paper considers a reverse link multiuser code division multiple access (CDMA) system with convolutional codes to protect against errors due to interference and thermal noise. Multiuser detection schemes [1] for uncoded data are shown to be extremely effective against multiple access interference. For coded information bits, the optimal detection and decoding strategy combines the trellises of all the users. However, the decoding complexity of this algorithm grows exponentially with the number of users [3]. A simple low-complexity alternative is to decouple the detection and decoding blocks. This reduced complexity algorithm suffers from significant performance loss. It is demonstrated that a tightly coupled pipelined iterative detection and decoding scheme can achieve the balance between performance loss and computational complexity.

The proposed iterative prior update (IPU) algorithm combines the maximum *a posteriori* (MAP) decoding rule with the successive interference cancellation scheme in an iterative framework. This basic idea of an iterative detection and decoding scheme has already been proposed by other researchers [4], [5]. The solution here differs in two aspects.

Manuscript received August 3, 2000; revised February 7, 2002; accepted August 7, 2004. The editor coordinating the review of this paper and approving it for publication is P. Driessen.

S. Das is with Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: sumand@bell-labs.com).

E. Erkip is with Polytechnic University, Brooklyn, NY 11201 USA (e-mail: elza@poly.edu).

J. R. Cavallaro and B. Aazhang are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: cavallar@rice.edu; aaz@rice.edu).

Digital Object Identifier 10.1109/TWC.2005.850314

First, *a posteriori* decision rule on entire code words rather than individual symbols is used. Also, instead of uniform priors, decoding results from previous iterations were used not only to reduce interference but also to update priors of decoder. We argue, with numerical examples, how this strategy improves over simpler joint detection and decoding schemes.

Second, a pipelined version of the IPU algorithm is proposed to reduce the decoding latency associated with the traditional word-based joint detection and decoding scheme. It is essentially a variant of the standard Viterbi algorithm [10] and hence can be implemented with currently available highly optimized decoders with minor modifications. We argue, with numerical examples, that for real-time applications this strategy provides a receiver structure that has low computational complexity, low delay and buffer size requirements, and thus can be deployed in practical systems.

After describing the system model in Section II, details of the IPU algorithm are provided in Section III along with discussions on computational complexity and delay issues. Numerical examples are provided in Section IV followed by concluding remarks.

II. System Model

Consider a reverse link system with K users, user k transmitting a block of N-coded bits $\mathbf{d}_k$ corresponding to the information bits $\mathbf{b}_k$. The unique repetitive spreading sequence s_k and the received signal strength A_k are assumed to be constant over this transmission block. The discretized chip-matched filter output at the receiver is given by [2] $\mathbf{r} = \mathbf{SAd} + \mathbf{z}$, where $\mathbf{S}$ and $\mathbf{A}$ are the matrix representations of the spreading sequence and the received signal strength, respectively, and $\mathbf{d} = [\mathbf{d}_1, \ldots, \mathbf{d}_K]^{\mathrm{T}}$ are the coded bits of all users. The additive white Gaussian noise vector is given by $\mathbf{z}$.

The code-matched filter output $\mathbf{y} = \mathbf{S}^{\mathrm{H}}\mathbf{r}$ is a sufficient statistic to estimate the transmitted symbols [1]. The MAP estimate of $\mathbf{d}$ (and, therefore, $\mathbf{b}$), given the matched filter output $\mathbf{y}$, minimizes the probability of sequence estimation error. Using the strict monotonicity of the logarithm function and Bayes' rule, we can write the log-MAP estimate as

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}\in\mathcal{C}} \log p(\mathbf{d}|\mathbf{y}) = \arg\max_{\mathbf{d}\in\mathcal{C}} \left[\log p(\mathbf{y}|\mathbf{d}) + \log p(\mathbf{d})\right] \tag{1}$$

where $\mathcal{C}$ denotes the collection of all the users' codebooks. The optimal code words $\mathbf{d}$ for all users are obtained by a joint search over the codebooks of all K users. The search

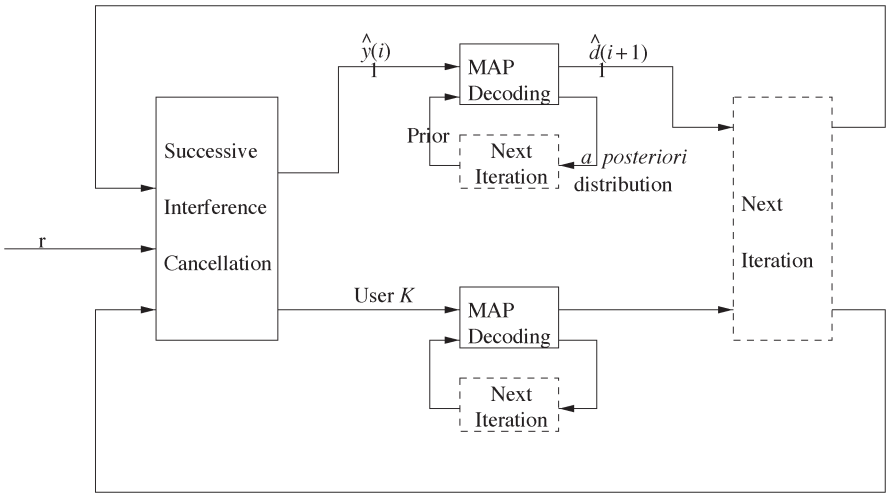

Fig. 1. Block diagram of the IPU algorithm.

space is exponential in number of users, and hence, this search algorithm is not suitable for real-time implementation. It is to be noted that the well-known maximum-likelihood sequence detection rule is a special case of (1) when there are uniform priors on the coded bit sequences of all the users. However, it is demonstrated that a MAP-based decoding strategy is more suitable for an iterative detection and decoding scheme.

III. Iterative Multiuser Detection and Decoding

Let $I_k = \{1, \ldots, k-1, k+1, \ldots, K\}$ represent the set of interferers for user k. The received signal can be rewritten as $\mathbf{r} = \mathbf{S}_k \mathbf{A}_k \mathbf{d}_k + \mathbf{S}_{I_k} \mathbf{A}_{I_k} \mathbf{d}_{I_k} + \mathbf{z}$, explicitly separating out the desired signal and the interference signal for user k. If there are accurate estimates of the coded bit sequences of the interferers $\hat{\mathbf{d}}_{I_k}$, then the signal $\hat{\mathbf{y}}_k = \mathbf{S}_k^{\mathrm{H}}(\mathbf{r} - \mathbf{S}_{I_k} \mathbf{A}_{I_k} \hat{\mathbf{d}}_{I_k})$ would be a sufficient statistic for the estimation of the coded bit sequence

$$\hat{\mathbf{d}}_k = \arg \max_{\mathbf{d}_k \in \mathcal{C}_k} \left[\log p(\hat{\mathbf{y}}_k | \mathbf{d}_k) + \log p(\mathbf{d}_k) \right]. \tag{2}$$

The higher the reliability of the decoded sequences $\hat{\mathbf{d}}_{I_k}$, the lesser is the residual interference in $\hat{\mathbf{y}}_k$ and the better is the estimate of $\mathbf{d}_k$ from $\hat{\mathbf{y}}_k$. In an efficient iterative detection and decoding scheme, the reliability of the decoded sequences $\hat{\mathbf{d}}_{I_k}$ improves with iterations. The proposed IPU algorithm is structurally similar to many other iterative detection and decoding schemes [6], [7] as illustrated in Fig. 1. During the ith iteration, for each user k, estimates of the code words of the interfering users $\hat{\mathbf{d}}_{I_k}$ are used from the previous step to obtain a semi-interference-free soft signal $\hat{\mathbf{y}}_k$. The single-user MAP decoding algorithm is then used to obtain the estimates $\hat{\mathbf{d}}_k$ that are subsequently used in the next iteration. However, unlike previous solutions, instead of a uniform prior, in each iteration the decoding process is used not only to estimate the coded bits $\mathbf{d}_k$ but also to compute and reuse the likelihood distribution of the entire code word space. In fact, in each iteration, posterior estimates from previous iterations are used as priors for the subsequent iteration. It bears noting that the MAP estimate is essentially the mode of this distribution. Initially, due to the low reliability of the decoded code block sequence, the mode by itself is not an accurate indicator of the posterior probability distribution $p(\mathbf{d}|\mathbf{y})$. Over iterations, the probability density function (pdf) will gravitate towards a unit mass centered at the mode of the distribution.

An effective real-time IPU algorithm should also address decoding delay and storage requirements. A simple variation of the Viterbi algorithm can be used to compute not only the mode but also the entire distribution $p(\mathbf{y}|\mathbf{d})$ and (1). Generally, this requires tracking 2^{NR} code word prefixes per user, where R is the coding rate, and is prohibitive in both computational and storage space requirements.

However, the *a posteriori* probability (APP) of a handful of code words can very well approximate the entire distribution. Thus, akin to list decoding [11], [12], the APP of top L code words is calculated, stored, and used. The sensitivity of performance to L is studied through simulations.

To reduce decoding delay for real-time applications, instead of a block decoding technique, a code word window of $\{1, \ldots, 5\kappa\}$, where κ is the constraint length of the convolutional code to decode the first symbol, was considered. In the next stage, we slide the window to include the $5\kappa + 1$th code symbol and drop the already decoded first symbol to decode the second symbol.

The pipelined IPU algorithm combines the above two ideas. For each of these decoding windows, the L most probable paths need to be computed. However, the top L paths up to level t in the trellis will not necessarily lead to the top L most probable paths up to any subsequent levels. Lemma 1 shows how to overcome this problem in a storage-efficient manner by maximizing the reuse of previous computations.

Lemma 1: For a convolutional code of constraint length κ, to evaluate the top L paths (with largest likelihoods) at any level or depth of the trellis, we need to store at most $L2^{\kappa}$ path metrics (see [13] for the details of the proof).

The proof of Lemma 1 demonstrates that having the L most probable paths terminating at each of nodes at the $(t-1)$th level of the trellis will enable calculating the top L most probable paths up to the tth level. There are 2^{κ} nodes at each level of the trellis for a convolutional code of constraint length κ, and hence we need to store at most $L2^{\kappa}$ paths.

The list decoding algorithm involves identifying the L most probable paths up to a certain level of the trellis. Naive implementation will involve elaborate sorting for each stage of the pipeline. Using clever data structure and exploiting the relationship between computations in each stage, the complexity can be dramatically reduced. Specifically, it is shown that if a sorted list of L most probable paths up to all nodes in level $t-1$ is maintained, then during the computation of the same metric for all nodes up to level t the repeated elaborate sorting can be avoided.

To compute the L most likely paths ending at node s_t at level t, the set of all the nodes $\mathcal{S}_{t-1}$ at level $t-1$ that have a branch ending at s_t is considered. The list of the top L paths for each of these nodes is stored in a descending order during the calculation in the pipeline stage. Now since for all paths ending at s_t and going through some node $s_{t-1} \in \mathcal{S}_{t-1}$, the log-probability of the path ending in s_t is the sum of the log probability of the path ending in s_{t-1} and the log-probability of the branch connecting s_{t-1} to s_t, and if a sorted list of the log-probabilities of all the paths terminating at s_{t-1} is maintained, then the paths ending at s_t and passing through s_{t-1} automatically get sorted. This is true for all nodes $s_{t-1} \in \mathcal{S}_{t-1}$. Thus, computation of the L most probable paths ending at s_t involves merging these sorted lists and choosing the top L elements from this merged list. Combining the above ideas, the steps of the delay and storage efficient IPU algorithm are presented next.

A. Delay and Buffer Efficient (DBE) IPU Algorithm

1) For all of the N-coded bits, set the initial coded sequence estimates $\hat{\mathbf{d}}_k = \mathbf{0}$ and initial priors $p(\mathbf{d}_k)$ uniform for $k \in \{1, \ldots, K\}$.
2) Begin with $n = 1$. In order to decode the nth-coded bit for all the users, take a block length of size 5κ bits, $[\hat{y}_k(n), \ldots, \hat{y}_k(n + 5\kappa - 1)]$. During estimation of the $(n-1)$th bits, we have computed the most probable code word sequence $[\hat{d}_k(n-1), \hat{d}_k(n), \ldots, \hat{d}_k(n + 5\kappa - 2)]$ and the log-probabilities of all code words ending at level $n + 5\kappa - 2$. The priors for the first iteration step of the current block are essentially the posterior probabilities of the code word sequences obtained from the decoding window for the $(n-1)$th bit and the initial estimate is $[\hat{d}_k(n), \ldots, \hat{d}_k(n + 5\kappa - 2), 0]$. Iterate as follows.
   a) In iteration step i, for user $k \in \{1, \ldots, K\}$
      i) Set $\hat{\mathbf{y}}_k^i = \mathbf{S}_k^{\mathrm{H}}(\mathbf{r} - \mathbf{S}_{I_k}\mathbf{A}_{I_k}\hat{\mathbf{d}}_{I_k}^{i-1})$, where $I_k = \{1, \ldots, k-1, k+1, \ldots, K\}$.
      ii) Set $\hat{\mathbf{y}}_k = \hat{\mathbf{y}}_k^i$ and the posterior probabilities from the $(i-1)$th iteration as $p(\mathbf{d}_k)$ in (2). Calculate the posterior probabilities and the estimate $\hat{\mathbf{d}}_k^i$ for the ith step. Use Lemma 1 to calculate the log-likelihoods for only the top L paths in the trellis. For all other paths, use a uniform probability that is smaller than the probability of the smallest among the L paths. Combine log-likelihoods with the prior to find the posterior.
      iii) Set the prior for the next iteration as $p(\mathbf{d}_k^{i+1}) = p(\mathbf{d}_k^i | \hat{\mathbf{y}}_k^i)$.
   b) Iterate until there is no further change in successive estimates for the first bit of the block for all the users.
3) Increase n by 1. If $n \leq N$, go back to step 2 to repeat the procedure to decode the next coded bit. Otherwise stop.

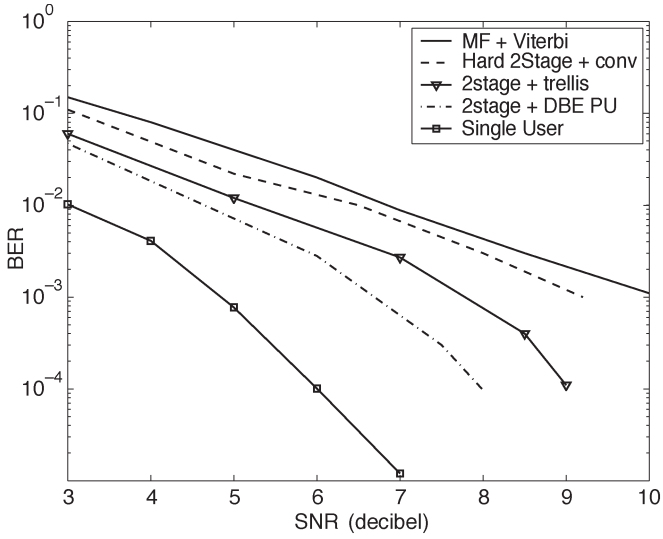

Fig. 2. Comparative study of various joint detection and decoding algorithms with a 12-user system and spreading gain of 31.

The algorithm requires $O[5\kappa(K + 2^{\kappa+1})]$ operations per coded bit per user per iteration, $O(5\kappa K)$ operations for interference cancellation, and $O(5\kappa 2^{\kappa+1})$ for the log-likelihood estimation. This value is slightly higher than the iterative algorithms proposed in the literature [3], which require $O(K + 2^{\kappa+1})$ operations per iteration. However, the algorithm significantly reduces the decoding delay to 5κ from N and the storage requirements to $2L2^{\kappa}$ from 2^{NR}. It is to be noted that to compute the path metrics for bit n, the path metrics for bit $n-1$ have to be remembered. This contributes to the multiplicative factor of 2 in the storage requirement expression. From the above analysis, it is clear that the primary difficulty in the real-time implementation of any joint detection and decoding scheme is associated with the complexity of the decoder. A simulation analysis of the proposed algorithm and a system example illustrating real-time decoding capabilities are provided next.

IV. Numerical Studies

A. Performance Analysis

Consider 12 user systems with a spreading gain of 31 (Gold codes) and the amplitude of all other users being twice that of user 1. For error protection, the users have a convolutional code of rate 2/3 and constraint length 5. Fig. 2 provides the bit error rate (BER) performance of user 1 for various multiuser detection/decoding algorithms.

The curve labeled "$\mathrm{MF} + \mathrm{Viterbi}$" is for a system where hard decisions on coded bits are made based on matched filter outputs and then passed through K single-user Viterbi decoders. "$\mathrm{Hard2stage} + \mathrm{conv}$" has a two-stage hard-output multistage detector followed by single-user Viterbi decoders. Since the multistage detector is better in mitigating multiple access interference, the performance of "$\mathrm{Hard2stage} + \mathrm{conv}$"

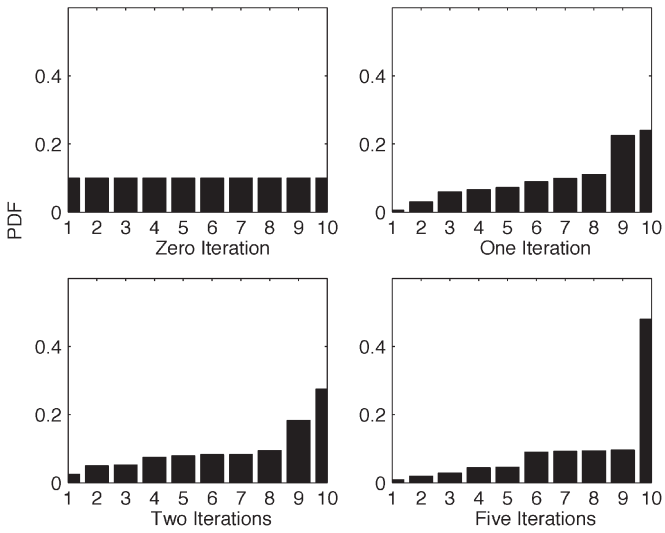

Fig. 3. Evolution of the normalized probability distribution of the top ten paths with the number of iterations.

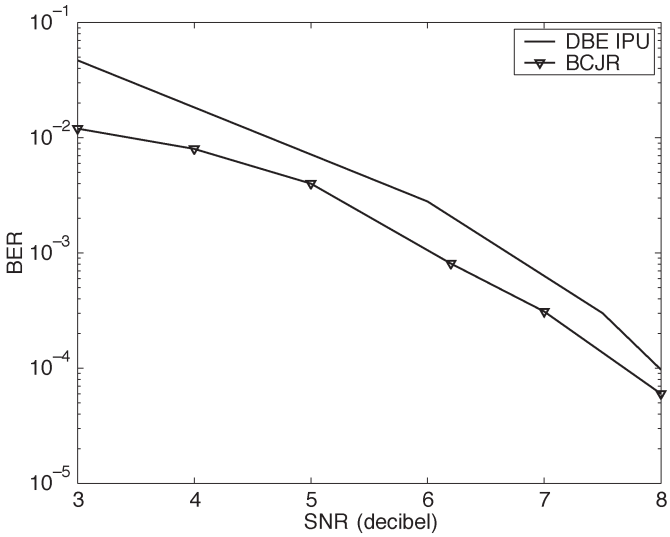

Fig. 4. Comparison of decoding algorithms based on BCJR algorithms (that minimize BER) and IPU decoding scheme for a 12-user system and spreading gain of 31.

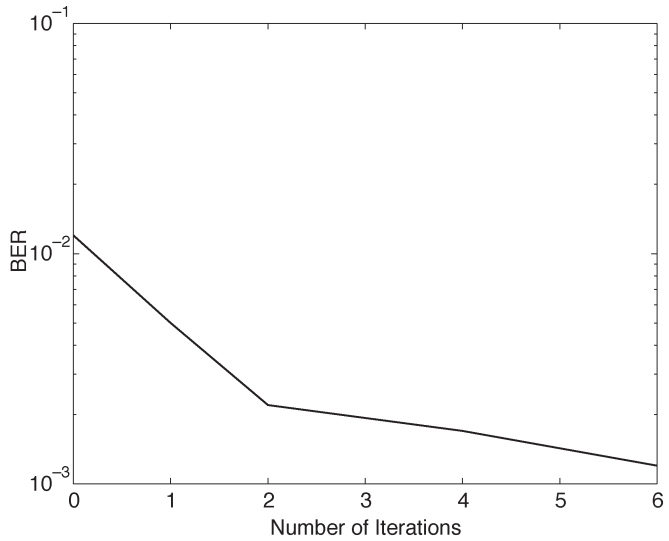

Fig. 5. Convergence study of the IPU algorithm; $K = 12$, $N_c = 31$, $L = 6$, convolutional code of rate $R = 2/3$, and $\kappa = 5$.

is superior to "$\mathrm{MF} + \mathrm{Viterbi}$." "$2\mathrm{stage} + \mathrm{trellis}$" refers to the algorithm described in [5]. A multistage detector in conjunction with K single-user soft Viterbi decoders is used. We have iterated the algorithm two times. Since this algorithm combines detection and decoding, performance is better than "$\mathrm{Hard2stage} + \mathrm{conv}$." Finally, "$2\mathrm{stage} + \mathrm{DBE\ IPU}$" refers to the DBE IPU algorithm described in Section III with two iteration steps. It was observed that the IPU algorithm consistently outperforms all the other schemes of comparable complexity. It provides a gain of about 0.5 dB over the best algorithm ("$2\mathrm{stage} + \mathrm{trellis}$") for a BER of 10^{-3}. It also comes to within 0.5 dB of the single-user bound for this loaded system. Instead of BER, if the block error rates of the above algorithms are considered, it was found that the relative performances of the algorithms remain unchanged.

Fig. 3 illustrates how the normalized probability distribution for the top ten paths $(L = 10)$ evolves over iterations. Within a few (two to three) iterations, it is possible to distinguish the most likely path with high reliability.

It is clear that further performance improvement over the IPU algorithm, which attempts to minimize the sequence error, can be achieved if algorithms that minimize the probability of symbol error are considered [6], [7]. However, traditionally Bahl, Cocke, Jelinek, and Raviv (BCJR) [9] based algorithms are an order of magnitude more complex (see [8] for details) and require larger decoding delays (due to its block-based nature) and storage.

The simulation comparison (Fig. 4) shows that the performance of this symbol MAP-based joint detection and decoding algorithm (labeled "BCJR") is better than the proposed IPU for low signal-to-noise ratio (SNR) ranges. However, for high SNR values, the performance difference is almost insignificant.

B. Cost of Implementation of the IPU Algorithm: Storage and Computational Requirements

Each iteration of the IPU algorithm improves the performance at an added computational cost. Fig. 5 illustrates the sensitivity of the performance of the IPU algorithm to the number of iterations. The simulation parameters are the same as in Fig. 2 and the SNR is fixed at 6 dB. We observe that IPU achieves its near optimal performance only after two to three iterations. Thus, the complexity of the algorithm, $O[5\kappa(K + 2^{\kappa+1})]$ per iteration, is essentially linear with the number of users.

The sensitivity of DBE IPU (with three iterations) to L, the number of paths stored, was also studied. By Lemma 1, L is directly related to the storage requirements of the system. The simulation results in Fig. 6 show that by storing probabilities for only a few (five to six) number of paths, the limiting performance can be achieved. Although both of the above parameters depend on the number of users in the system, the spreading gain N_c, and the constraint length of the code κ, the analysis shows that this dependence is quite weak. In fact, for

most systems with realistic spreading gains and with number of users as that simulated here, the number of paths needed to achieve the limiting performance was less than ten. Hence, as claimed, the algorithm has low storage requirements.

C. Real-Time Implementation

To illustrate the real-time capabilities of the IPU algorithm, consider a system with 15 users each transmitting at 20 kb/s. Each user has a convolutional code of rate 2/3 and constraint length 5. If a block length of 1024 data bits is considered, the receiver will have 0.05 s to decode all the information bits of all the users.

For the optimal joint multiuser trellis decoding technique, the complexity of the algorithm per user is given by $(N2^{K+\kappa+1})/K$. Using $N = 1024$, $K = 15$, and $\kappa = 5$ as above, a total of 128×10^6 operations per user is needed. Assuming a 200-MHz TI DSP processor (TMS320C6701) per user, it will require 0.64 s to complete all the operations. Even if the processor can utilize all the eight available functional units all the time, which in most applications is not possible, 0.08 s would still be needed.

For the DBE IPU algorithm, we have a smaller decoding delay, which is equal to 25 (5κ) bit periods (i.e., only 1.25 ms). In each pipeline stage, one new bit is pushed in and 1 bit is decoded. Consequently, since a data rate of 20 kb/s is considered, we would like to find out whether we can complete one stage of the pipeline and decode 1 bit in 0.05 ms. Recall from Section III that the DBE IPU algorithm requires $5\kappa(K + 2^{\kappa+1})$ operations per user per bit per iteration. It is shown in Fig. 5 that about two to three iterations are enough to get near optimal performance. Hence, for the above code and processor, assuming three iterations, we calculate that the DBE IPU algorithm can be completed in 30 μs. If we assume a very realistic two-way parallelism (two out of eight possible functional units used on the average), the decoding time per user per bit is only 15 μs, which is below the required 50 μs. Also, we need less than 1 kb of storage space for the partially decoded paths per user. Coupled with the fact that performance is better than other low-complexity multiuser schemes, we argue that the IPU algorithm is an attractive alternative for real-time implementations.

V. Conclusion

In this paper, a low-complexity multiuser joint detection and decoding algorithm for a convolutionally coded direct sequence code division multiple access (DS-CDMA) system has been described. The main goal was to show that this algorithm is suitable for systems requiring real-time communications such as cellular voice. The optimal joint detector/decoder has exponential complexity in the number of mobile users and is not practical for such real-time applications. The proposed iterative prior update (IPU) algorithm is based on iterative interference cancellation together with maximum *a posteriori* (MAP) decoding for sequences of coded bits and prior updates at every iteration. It requires a small storage space for storing priors along iterations, has low decoding delay, and has fast convergence. The complexity can be shown to be linear with the number of users. It also has the added benefit of utilizing the already existing hardware for Viterbi decoding. Through simulations, it is shown that the performance is superior to other low-complexity sequence detection strategies. A numerical example illustrating the benefits of the algorithm in real-time applications is also provided.

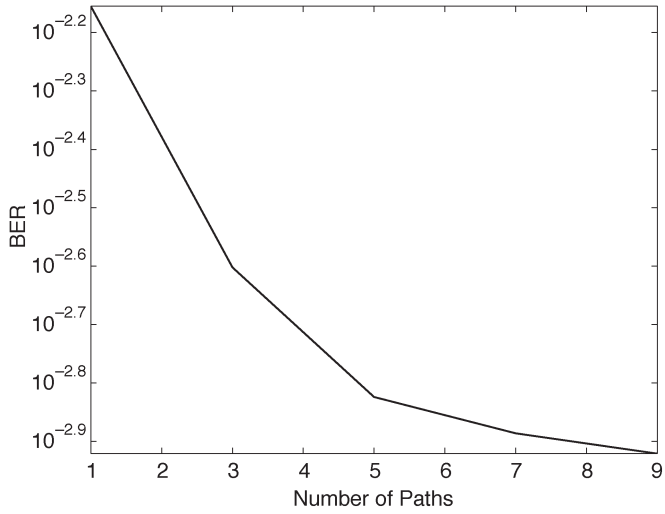

Fig. 6. Sensitivity study of the DBE IPU algorithm to storage space, $K = 12$, $N_c = 31$, $R = 2/3$, and $\kappa = 5$.

References

[1] S. Verdú, *Multiuser Detection*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[2] J. G. Proakis, *Digital Communications*. New York: McGraw-Hill, 1995.
[3] T. R. Giallorenzi and S. G. Wilson, "Multiuser ML sequence estimator for convolutional coded asynchronous system," *IEEE Trans. Commun.*, vol. 44, no. 8, pp. 997–1008, Aug. 1996.
[4] ——, "Suboptimum multiuser receivers for convolutionally coded asynchronous DS-CDMA systems," *IEEE Trans. Commun.*, vol. 44, no. 9, pp. 1183–1196, Sep. 1996.
[5] U. Fawer and B. Aazhang, "Multiuser receivers for code-division multiple-access systems with trellis-based modulation," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 8, pp. 1602–1609, Oct. 1996.
[6] P. D. Alexander, M. C. Reed, J. A. Asenstorfer, and C. B. Schlegel, "Iterative multiuser interference reduction: Turbo CDMA," *IEEE Trans. Commun.*, vol. 47, no. 7, pp. 1008–1014, Jul. 1999.
[7] X. Wang and H. V. Poor, "Iterative (turbo) soft interference cancellation and decoding for coded CDMA," *IEEE Trans. Commun.*, vol. 47, no. 7, pp. 1046–1061, Jul. 1999.
[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," in *Proc. Int. Conf. Communications (ICC)*, Geneva, Switzerland, 1993, pp. 1064–1070.
[9] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.
[10] H. Hendrix, "Viterbi Decoding Techniques in the TMS320C54x Family," *Texas Instruments Application Report*, 1996, Dallas, TX: Texas Instruments. [Online]. Available: http://www-s.ti.com/sc/psheets/spra071/spra071.pdf
[11] K. R. Narayanan and G. L. Stuber, "List decoding of turbo codes," *IEEE Trans. Commun.*, vol. 46, no. 6, pp. 754–762, Jun. 1998.
[12] J. B. Anderson and S. Mohan, "Sequential coding algorithms: A survey and cost analysis," *IEEE Trans. Commun.*, vol. COM-32, no. 2, pp. 169–176, Feb. 1984.
[13] S. Das, E. Erkip, J. R. Cavallaro, and B. Aazhang, "Maximum weight basis decoding of convolutional codes," in *Proc. IEEE Global Telecommunication Conf. (Globecom)*, San Francisco, CA, Nov. 2000, vol. 2, pp. 835–841.

Suman Das (S'03–M'04) received the B.S. degree in computer science and engineering from the Indian Institute of Technology, Kharagpur, India, in 1994, and the M.S. and Ph.D. degrees from Rice University, Houston, TX, in 1997 and 2000, respectively.

Since 2000, he has been a Member of Technical Staff at the Wireless Technology Research Department, Bell Laboratories, Lucent Technologies, Murray Hill, NJ. His primary research interests include algorithms and architectures for signal processing and communication and design and analysis of future wireless networks.

Elza Erkip (S'93–M'96) received the B.S. degree in electrical and electronic engineering from the Middle East Technical University, Ankara, Turkey, in 1990, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1996 and 1993, respectively.

From 1996 to 1999, she was with the Department of Electrical and Computer Engineering of Rice University. In Spring 2000, she joined the Polytechnic University, where she is currently an Assistant Professor of Electrical and Computer Engineering.

Joseph R. Cavallaro (S'79–M'82–SM'05) was born in Philadelphia, PA. He received the B.S. degree in electrical engineering from the University of Pennsylvania, Philadelphia, PA, in 1981, the M.S. degree in electrical engineering from Princeton University, Princeton, NJ, in 1982, and the Ph.D. degree in electrical engineering from Cornell University, Ithaca, NY, in 1988.

From 1981 to 1983, he was with AT&T Bell Laboratories, Holmdel, NJ. In 1988, he joined the faculty of Rice University, Houston, TX, where he is currently a Professor of Electrical and Computer Engineering. From the 1996 to 1997 academic year, he served at the U.S. National Science Foundation as Director of the Prototyping Tools and Methodology Program in the Computer (CISE) Directorate. In 2005, he was a Nokia Foundation Fellow and a Visiting Professor at the Centre for Wireless Communications at the University of Oulu, Finland. He is currently the Associate Director of the Center for Multimedia Communication at Rice University. His research interests include computer arithmetic, very large scale integration (VLSI) design and microlithography, and digital signal processor (DSP) and VLSI architectures for applications in wireless communications.

Dr. Cavallaro was the recipient of the National Science Foundation (NSF) Research Initiation Award and is a Member of Tau Beta Pi, Eta Kappa Nu, and a Senior Member of the IEEE. He is an IEEE Computer Society Distinguished Lecturer from 2004 to 2006, and was Co-Chair of the 2004 Signal Processing for Communications Symposium at the IEEE Global Communications Conference and General Co-Chair of the 2004 IEEE 15th International Conference on Application-specific Systems, Architectures and Processors (ASAP).

Behnaam Aazhang (S'82–M'82–SM'91–F'99) received the B.S. (with highest honors), M.S., and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana–Champaign in 1981, 1983, and 1986, respectively.

From 1981 to 1985, he was a Research Assistant at the Coordinated Science Laboratory, University of Illinois. In August 1985, he joined the faculty of Rice University, Houston, TX, where he is now the J.S. Abercrombie Professor and Chair of the Department of Electrical and Computer Engineering and also the Director of Center for Multimedia Communications. He has been a Visiting Professor at IBM Federal Systems Company, Houston, TX, at the Laboratory for Communication Technology, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, at the Telecommunications Laboratory, University of Oulu, Oulu, Finland, at the U.S. Air Force Phillips Laboratory, Albuquerque, NM, and at Nokia Mobile Phones in Irving, Texas. His research interests are in the areas of communication theory, information theory, and their applications, with emphasis on multiple access communications cellular mobile radio communications and wireless communication networks.

Dr. Aazhang is a Fellow of IEEE, the recipient of the Alcoa Foundation Award in 1993, the National Science Foundation (NSF) Engineering Initiation Award from 1987 to 1989, and the IBM Graduate Fellowship from 1984 to 1985, and is a Member of Tau Beta Pi and Eta Kappa Nu. He is also a recipient of 2004 IEEE Communication Society's Stephen O. Rice best paper award for a paper with A. Sendonaris and E. Erkip. He has served on Houston Mayor's Commission on Cellular Towers from 1998 to 2004, as the Editor for Spread Spectrum Networks of IEEE TRANSACTIONS ON COMMUNICATIONS from 1993 to 1998, the Treasurer of IEEE Information Theory Society from 1995 to 1998, the Secretary of Information Theory, San Antonio, TX, and as Co-Chair of the Technical Program Committee of the 2001 Multi-Dimensional and Mobile Communication (MDMC) Conference in Pori, Finland. He will be serving as the Chair of the Technical Program Committee for the 2005 Asilomar Conference of International Workshop on Convergent Technologies (IWCT), Oulu, Finland, June 6–10, 2005.